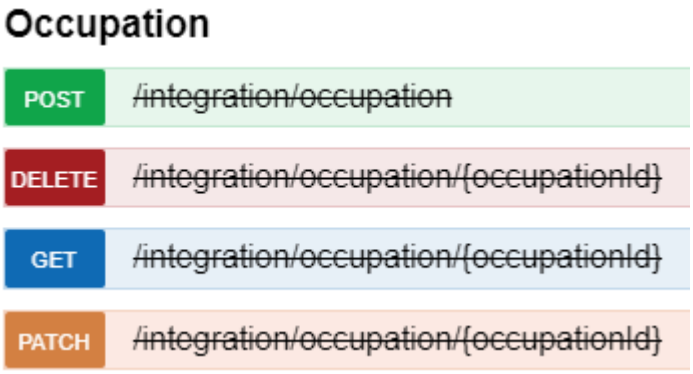
CoreAPI - Starter Package
Documentation fonctionnelle de base

Informations générales

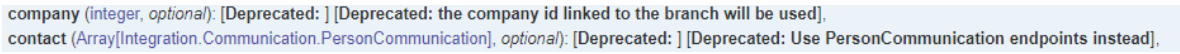
Points de terminaison et attributs obsolètes

Comme de nouvelles fonctionnalités sont prévues régulièrement, il n'est pas possible de continuer à supporter indéfiniment tous les points de terminaison et attributs existants. Pour indiquer les points de terminaison et les attributs qui finiront par disparaître, le terme "obsolète" est utilisé.

Via Swagger, les points de terminaison obsolètes sont clairement indiqués en les barrant.
Exemple.



Les attributs obsolètes peuvent être trouvés dans Swagger dans les modèles de réponse et de demande.
Exemple.

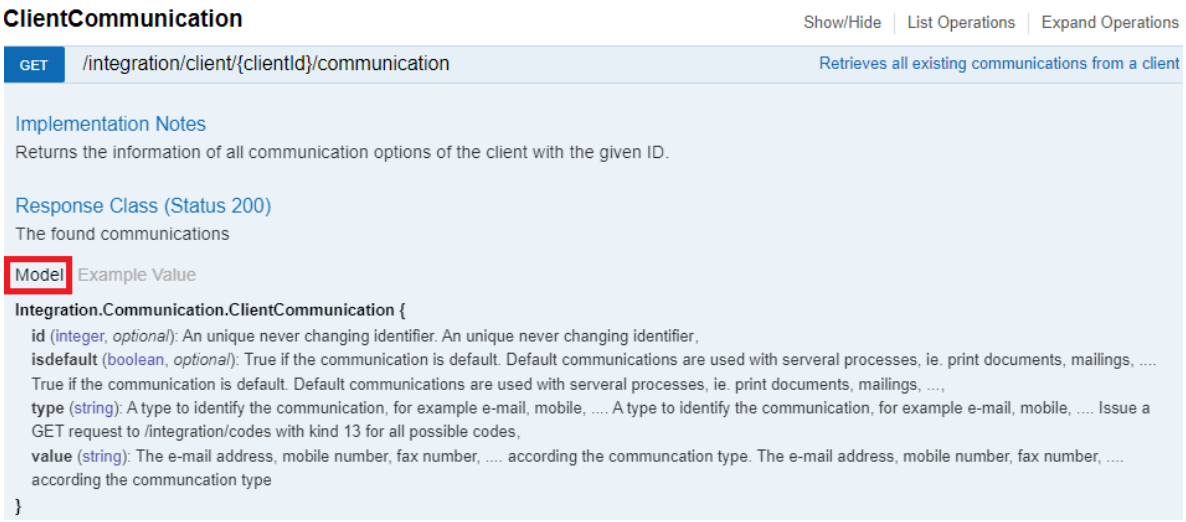


Vous trouverez de plus amples informations en cliquant sur les liens suivants (NL et FR)
Un aperçu des points de terminaison/attributs dépréciés et des alternatives possibles peut être trouvé via cette page

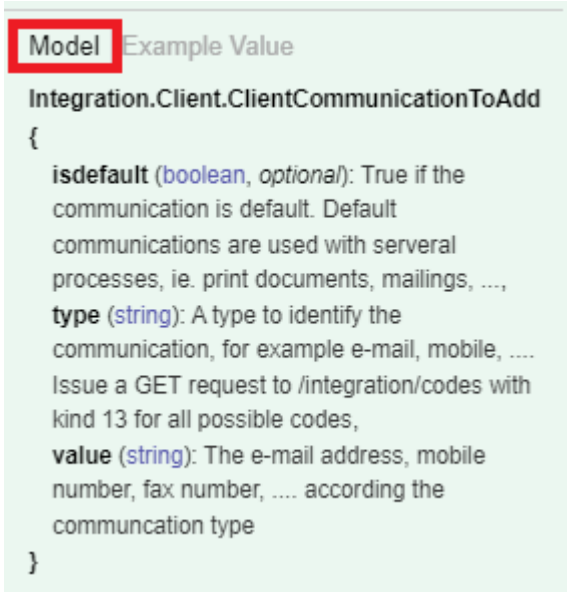
Modèles de données de la réponse et de la demande

Via Swagger, vous pouvez trouver plus d'informations sur les modèles de données de demande et de réponse. Pour chaque attribut, il y a une brève description fonctionnelle et des références à d'autres points de terminaison, le cas échéant.

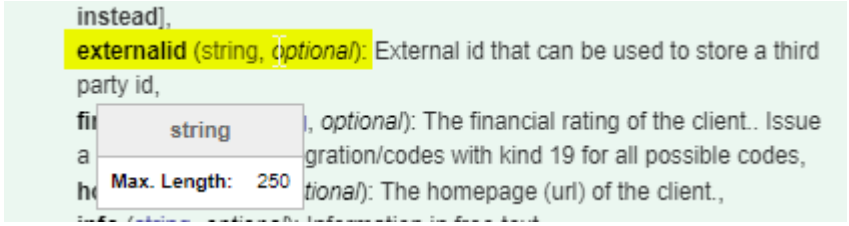
Cliquez sur “Model” sur le modèle de réponse, par ex. :



Ou sur le modèle de demande, par ex. :



Il est possible de survoler certains attributs (que nous continuons à enrichir) pour afficher des informations supplémentaires.
par ex. la longueur max.



Authentification avec un token

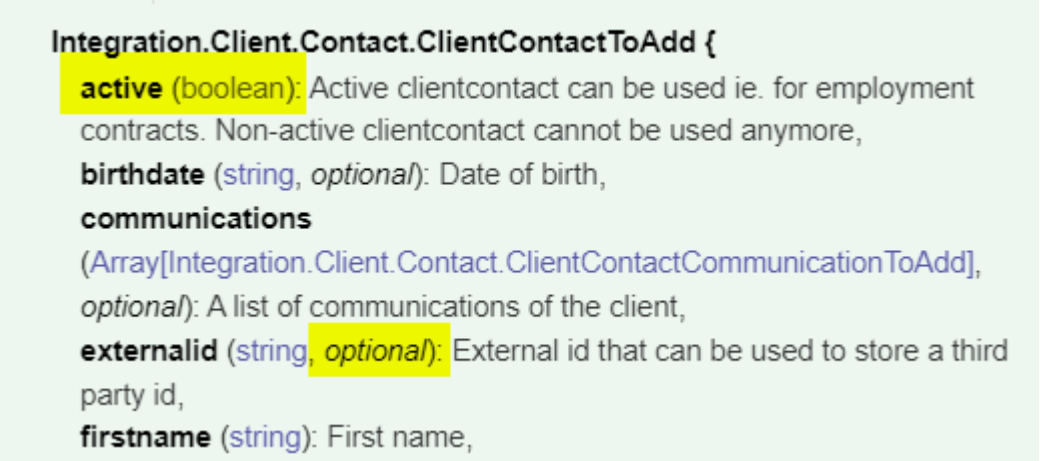
Lors de l'installation de l'API, un token d'authentification est également transmis. Ce token, ainsi que le string "WB", doivent être transmis lors de chaque appel vers l'API. Par exemple:

```
curl -X GET --header 'Accept: application/json' --header 'Authorization: WB
ABC123JESUISUNTOKEN'
'http://<url>/webservice/integration/codes?kind=1000&language=fr'
```

En principe, un token pour l'API n'expire pas, bien qu'un nouveau token puisse être généré sur simple demande si nécessaire.

Quelles sont les données obligatoires dans le modèle de demande?

Vous pouvez utiliser Swagger pour savoir, pour chaque attribut, si la donnée est obligatoire ou non.
Les attributs indiqués comme "optional" ne sont pas obligatoires
Les attributs sans indication sont obligatoires.

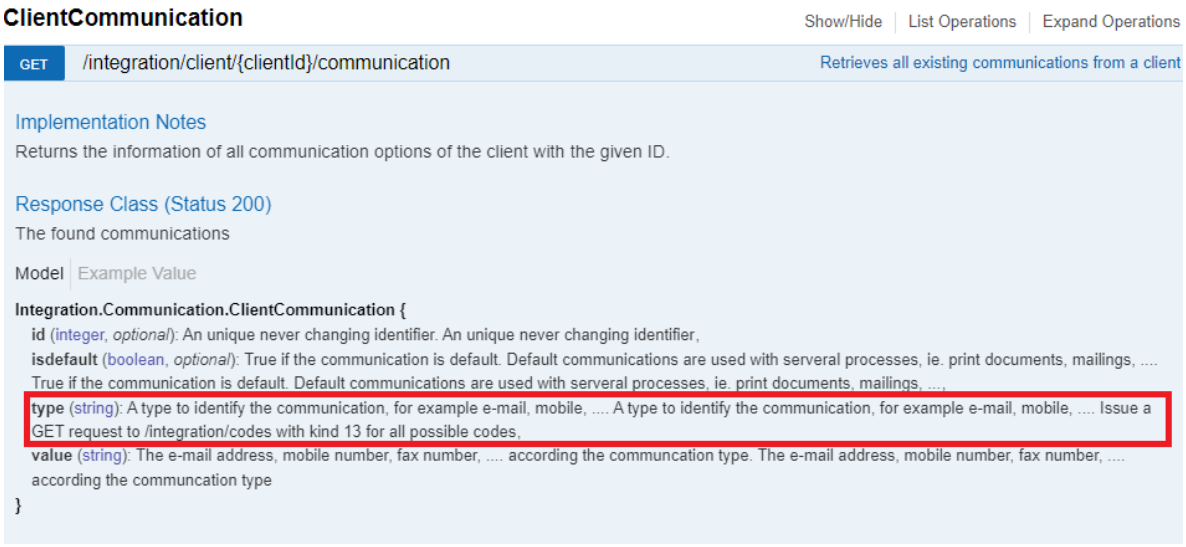


Codes et descriptions

Les données dans PratoFlex sont souvent modelées avec des codes génériques. Cela signifie que l'agence intérimaire elle-même peut créer, modifier et supprimer de nouveaux codes pour certaines listes.

Les modèles de réponse de l'API ne renvoient que la valeur du code. Si la description est requise, elle peut être demandée via le point de terminaison **GET /integrations/codes**.

Par exemple, si vous voulez récupérer les canaux de communication d'un client par l'intermédiaire de **GET /integration/client/{clientId}/communication**, alors vous remarquerez dans Swagger que ce type de données est liée au type de code 13:



Vous pouvez alors utiliser le GET suivant pour récupérer les codes appartenant au type de code 13:

https://<url>/webservice/integration/codes?kind=13&language=fr
(language=nl pour les descriptions en néerlandais)

En réponse, vous recevrez une liste de toutes les options qui appartiennent au type 'canaux de communication' du client.


```
[
  {
    "description": "Téléphone",
    "descriptionshort": null,
    "externalcode": null,
    "id": "1",
    "kind": "13",
    "language": "fr",
    "validfrom": null,
    "validuntil": null
  },
  {
    "description": "Fax",
    "descriptionshort": null,
    "externalcode": null,
    "id": "2",
    "kind": "13",
    "language": "fr",
    "validfrom": null,
    "validuntil": null
  }, etc
```


Fonctionnement d'un PATCH

Pour tous les appels PATCH à l'api qui se produisent, nous supposons que **seuls** les champs qui doivent être modifiés sont envoyés. Tous les champs qui ne sont pas inclus conserveront leur valeur initiale.

Réponses HTTP

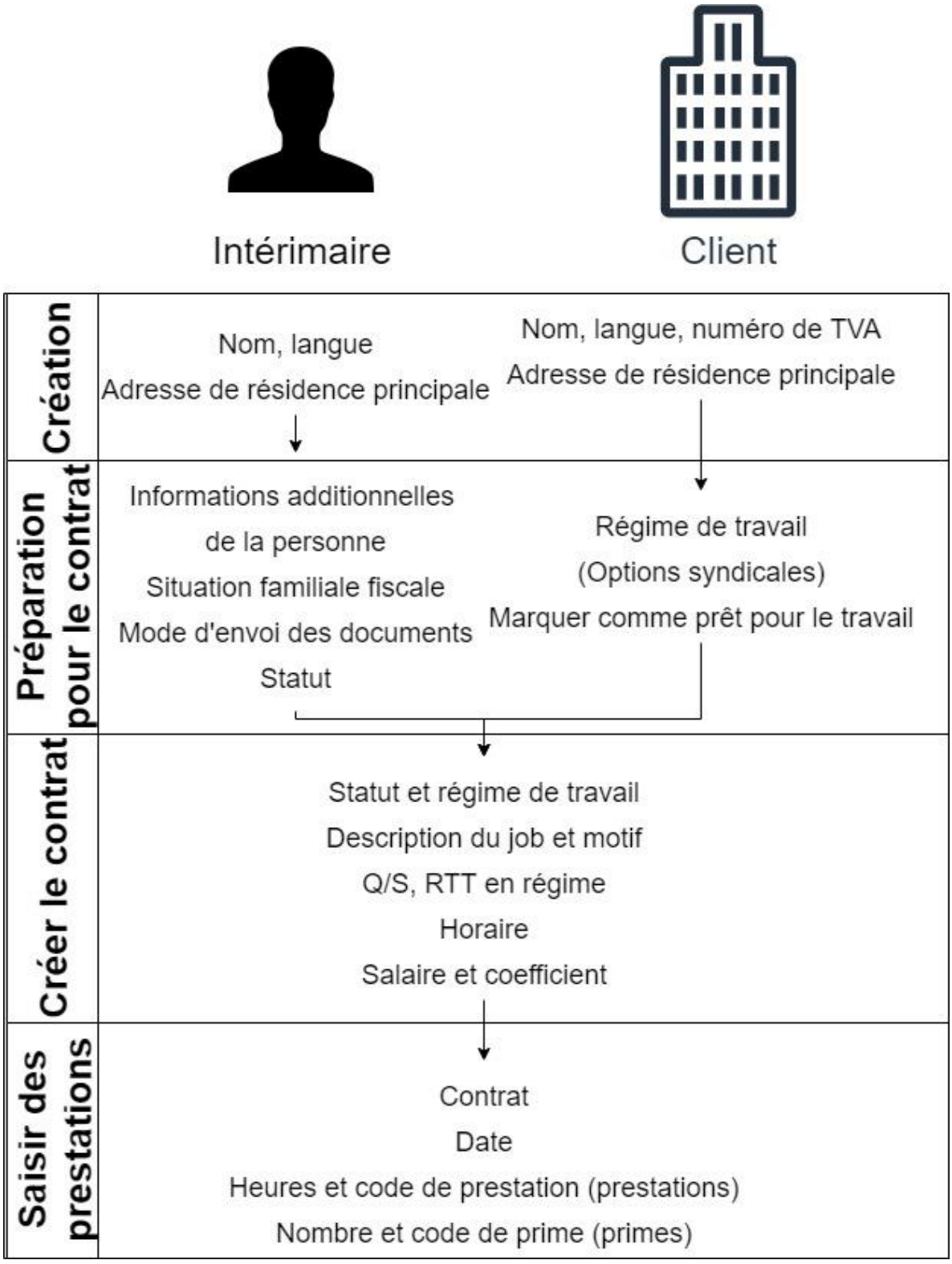
Nous utilisons les réponses HTTP générales selon, par exemple, les critères suivants
https://developer.mozilla.org/en-US/docs/Web/HTTP/Status

Les plus couramment utilisées sont:

- 200 OK
- 201 Created
- 202 Accepted : Voir également Webhook, callback urls
- 204 No Content : Les points de terminaison DELETE renvoient ce message
- 400 Bad Request : Un message d'erreur (fonctionnel) est renvoyé avec la réponse. Le message d'erreur devrait dans la plupart des cas expliquer ce qui doit être corrigé. Souvent, les messages d'erreur peuvent être interprétés par un utilisateur qui ajuste les données.
- 401 Unauthorized : Le token est-il correct? voir également Authenticatie avec un token
- 404 Not Found : Les identifiants sont-ils corrects?
- 500 : Une erreur système s'est produite (par ex. l'API est indisponible en raison d'une mise à jour, mais aussi de messages d'erreur inattendus). **Ces demandes peuvent être relancées par un mécanisme de réessai, par exemple 5 fois. Si ces messages d'erreur ne sont pas résolus, il est nécessaire de le signaler via customercare@prato.be**

Use cases

Flux de base



Créer une personne

Vous pouvez créer une personne via **POST /integration/person**. Pour créer une personne, seuls quelques champs sont obligatoires (voir Swagger), mais vous pouvez bien sûr fournir beaucoup plus d'informations lors de la création. Une personne doit également disposer d'une adresse de résidence principale. Vous pouvez créer une personne avec le body suivant:


```
{
"branch": 1,
"firstname": "Exemplepersonne1",
"name": "Prato",
"language": 2,
 "addresses": [
  {
   "addresstype": 1,
   "city": "Hasselt",
   "country": 150,
   "street": "Kempische Steenweg",
   "housenumber": "293/35",
   "validfrom": "2022-01-01T00:00:00.000Z",
   "zipcode" : "3500"
  }
 ]
}
```


Remarque : Le type d'adresse "domicile" peut être différent de la valeur 1 dans certains environnements (<5%), cela peut arriver parce que différents codes ont été définis pour le client. Le code correct pour l'environnement peut être récupéré via le point de terminaison GET /integration/codes avec le paramètre kind=86.

Créer un client

Vous pouvez créer un client via **POST /integration/client**. Pour créer un client, seuls quelques champs sont obligatoires (voir Swagger), mais vous pouvez bien sûr fournir beaucoup plus d'informations lors de la création. Un client doit également avoir une adresse de type 100 (une adresse de résidence principale). Avec le body suivant, vous pouvez créer un client:


```
{
 "addresses": [
  {
   "city": "Hasselt",
   "country": 150,
   "housenumber": "2G",
   "street": "Gouverneur Verwilghensingel",
   "type": 100,
   "zipcode": "3500"
  }
 ],
 "branch": 1,
 "language": "1",
 "name": "Exempledeclient",
 "taxnumber" : "BE0454896841"
}
```


Remarque : contrairement aux types d'adresse d'une personne, l'adresse de résidence principale d'un client est toujours la valeur 100.

Compléter les données d'une personne pour créer des contrats

Pour pouvoir créer un contrat pour une personne, il faut disposer d'informations supplémentaires relatives au contrat de cette personne.

Date de naissance, nationalité et NISS

Avant de pouvoir créer un statut, certaines informations personnelles supplémentaires habituelles sont également requises (date de naissance, nationalité, NISS).

L'information pourrait aussi être donnée via le POST, si elle est ajoutée plus tard, cela peut se faire via le **PATCH /integration/person**.

```
{
   "inss" : "95072500528",
   "nationality" : "150",
   "birthdate" : "1995-07-25"
}
```

Infos fiscales

Des informations sur la situation familiale fiscale de la personne doivent être fournies. Cette information peut être donnée via **POST /integration/person/{persionid}/taxationhistory**. Vous pouvez y saisir des données telles que l'état civil, le nombre d'enfants à charge, etc. Via le body suivant, nous indiquons que notre personne de test est célibataire et nous indiquons 1 enfant à charge:

```
{
   "maritalstate" : "1",
   "validfrom": "2022-07-20",
   "childrenatexpensenumber": 1
}
```

Mode d'envoi des documents

La méthode d'envoi des documents pour une personne doit être indiquée. Cette information peut être fournie via **PATCH /integration/person**. Par l'intermédiaire du body suivant, nous avons configuré une adresse e-mail comme méthode d'envoi pour notre personne exemple:

```
{
   "deliverymethoddocuments": {
      "deliverymethodtype": "1",
      "mailto": "personne.exemple@prato.be"
   }
}
```

Remarque : il se peut que l'environnement du client soit configuré de telle sorte que le courrier électronique et l'envoi via Unified Post ne soient pas possibles. Cela peut être le cas notamment pour les environnements de test. Ces environnements n'attendent aucune action concernant le mode d'envoi, mais nous conseillons tout de même de toujours prévoir ce call car les environnements de production sont souvent configurés avec e-mail et Unified Post.

Statut

Une personne a également besoin d'un statut avant de pouvoir établir un contrat. Cette information peut être fournie via **PATCH /integration/{personid}/statute**. Avec cela, vous donnez une catégorie (ouvrier, employé, étudiant, flex), et une période de paiement (hebdomadaire, mensuel). Via

taxtypeinternal, vous configurez la manière dont le pourcentage du PP doit être déterminé. Avec le body suivant, nous donnons un statut d'emploi à notre personne exemple.


```
{
   "branchid": 1,
   "section": "115",
   "taxtypeinternal": 1
}
```


Remarque : Veuillez noter que le code pour ce statut peut différer d'un environnement à l'autre. Les valeurs pour le client peuvent être récupérées via GET /integration/codes avec kind=96.

Compléter un prospect/client afin de pouvoir créer des contrats

Afin de créer un contrat, certaines informations du client sont requises.

Régime de travail

Dans le régime de travail d'un client, vous fournissez des données telles que la catégorie d'employé, la commission paritaire, le nombre d'heures RTT payées/non payées, le nombre d'heures travaillées en cas de temps plein, etc. Cela peut être défini via **POST** ou **PATCH /integration/client/{clientid}/workregime**. Via le body suivant, nous configurons le régime de travail d'un ouvrier, commission paritaire 125, un RHM (=régime horaire moyen) de 38, RHR (régime horaire réel) de 40 avec RTT (réduction du temps de travail) rémunéré.


```
{
  "averagetotalworkhours": 38,
  "category": "1",
  "description": "Ouvrier 38/40 RTT payé",
  "jointcommittee": "125",
  "realtotalworkhours": 40,
  "validfrom": "2022-07-20",
  "workingsystem": "5",
  "worktimereductionfulltime": "paid",
  "worktimereductionparttime": "paid"
}
```


Marquer le client comme prêt pour le travail

Avant de pouvoir attribuer des contrats à un client, celui-ci doit être marqué comme prêt pour le travail. Cela peut être fait via **PUT /integration/client/{clientid}/setreadyforemploymentcontracts**. Aucun body n'est requis pour ce PUT.

Créer un contrat

Vous pouvez créer un contrat via **POST /integration/employmentcontract**. Il est obligatoire de spécifier de quelle personne et de quel statut il s'agit, ainsi que le régime de travail du client. En outre, vous devez également préciser l'horaire et les informations telles que le salaire. Dans le body suivant, nous créons un contrat pour la personne exemple avec le client exemple.

{{clientId}} = l'id du client créé dans les étapes précédentes
{{employeeId}} = l'id du statut (**pas** de la personne) créé dans les étapes précédentes
{{workregimeId}} = l'id du régime de travail créé pour le client dans les étapes précédentes


```
{
  "branch": 1,
```

```
  "begindate": "2022-07-25",
  "enddate": "2022-07-31",
  "clientid": {{clientId}},
  "statuteid": {{employeeid}},
  "jobdescriptionid" : "409",
  "hourlywage": 12.39,
  "coefficient": 1.91,
  "selectiontempworker": "1",
  "remunerationmethod": "0",
  "type": "0",
  "turnoverbranch" : "1",
  "reasonofemployment": "2",
  "meansoftransport": "2",
  "clientworkregimeid": {{workregimeId}},
  "workingsystem": 5,
  "q": 40.00,
  "s": 40.00,
  "weeklyhoursfulltime": 38.00,
  "weeklyhoursparttime": 0.00,
  "worktimereductionpaid": 2.00,
  "worktimereductionunpaid": 0.00,
  "workschedule": {
   "monday": [
    {
     "timeworked": "8"
    }
   ],
   "tuesday": [
    {
     "timeworked": "8"
    }
   ],
   "wednesday": [
    {
     "timeworked": "8"
    }
   ],
   "thursday": [
    {
     "timeworked": "8"
    }
   ],
   "friday": [
    {
     "timeworked": "8"
    }
   ]
  }
}
```

Saisir des prestations et des primes

Pour les contrats, vous pouvez saisir des prestations et des primes.

Les prestations sont le nombre réel d'heures travaillées/prestées par jour. Ces nombres peuvent s'écarter des heures du contrat, soit en moins, soit en plus.

Les primes sont des indemnités supplémentaires (par exemple, 1€ de plus par jour travaillé, 10€ de prime de poste, 15€ de prime de travail de nuit, ...).

Saisir des prestations

Vous pouvez utiliser **POST /integration/employmentcontract/{employmentcontractid}/performances** pour soumettre des prestations pour un contrat. Le code de prestation, la date et le nombre d'heures sont obligatoires. Optionnellement, vous pouvez également inclure le centre de frais, le poste et la division. Dans l'exemple ci-dessous, les prestations sont saisies pour 3 jours:

Les valeurs possibles pour le *code* peuvent être trouvées via GET /integration/codes avec *kind=2000*


```
[
  {
    "code": "AD",
    "date": "2022-07-25",
    "hours": 8.0
  },
   {
    "code": "AD",
    "date": "2022-07-26",
    "hours": 8.0
  },
  {
    "code": "AD",
    "date": "2022-07-27",
    "hours": 8.0
  }
]
```


Saisir des primes

Vous pouvez utiliser **POST /integration/employmentcontract/{employmentcontractid}/premiums** pour soumettre des primes pour un contrat. Le code de la prime, le nombre de primes et la date sont obligatoires. Optionnellement, vous pouvez également saisir le montant, le centre de frais, le poste et la division. Dans l'exemple ci-dessous, les primes pour 3 jours sont saisies:

Les valeurs possibles pour le *code* peuvent être trouvées via GET /integration/codes avec *kind=1000*


```
[
  {
    "code": "201",
    "date": "2022-07-25",
    "number": 1,
    "amount": 10
  },
   {
    "code": "201",
    "date": "2022-07-26",
    "number": 1,
    "amount": 10
  },
  {
    "code": "201",
    "date": "2022-07-27",
    "number": 1,
    "amount": 10
  }
]
```

Queues

Il est possible de traiter les modifications apportées dans PratoFlex aux personnes, aux clients et aux contrats via une queue.

par ex. Un autre système maîtrise les données nom et adresse d'une personne, mais pas les données fiscales (par exemple, le nombre d'enfants à charge). L'autre système est intéressé par les données fiscales.

- L'autre système utilisera l'API pour tenir à jour le nom et l'adresse d'une personne
- Si un consultant modifie le nombre d'enfants à charge via PratoFlex, PratoFlex publiera l'identifiant de la personne dans une queue
- L'autre système peut aller chercher l'identifiant dans la queue et récupérer les nouvelles données via l'API
- Les nouvelles données fiscales peuvent alors également être enregistrées dans l'autre système

Si vous souhaitez obtenir plus d'informations à ce sujet, veuillez contacter le Customer Success Manager du client ou customercare@prato.be